
Digital Logic Design
ECEN 3233
Module 9b – FSM Optimization: State
Assignment and FSM Partitioning

Weihua Sheng
School of Electrical and Computer Engineering
Oklahoma State University

Spring 2007

State assignment

- Choose bit vectors to assign to each “symbolic” state
 - with n state bits for m states there are $2^n! / (2^n - m)!$
[$\log n \leq m \leq 2^n$]
 - 2^n codes possible for 1st state, $2^n - 1$ for 2nd, $2^n - 2$ for 3rd, ...
 - huge number even for small values of n and m
 - intractable for state machines of any size
 - heuristics are necessary for practical solutions
 - optimize some metric for the combinational logic
 - size
 - speed (depth of logic and fanout)

State assignment possibilities for 4 states

S0	S1	S2	S3
00	01	10	11
00	01	11	10
00	10	01	11
00	10	11	01
00	11	01	10
00	11	10	01
01	00	10	11
01	00	11	10
01	10	00	11
01	10	11	00
01	11	00	10
01	11	10	00

S0	S1	S2	S3
10	00	01	11
10	00	11	01
10	01	00	11
10	01	11	00
10	11	00	01
10	11	01	00
11	00	01	10
11	00	10	01
11	01	00	10
11	01	10	00
11	10	00	01
11	10	01	00

State assignment strategies

- Possible strategies
 - sequential – just number states as they appear in the state table
 - random – pick random codes
 - one-hot – use as many state bits as there are states (bit=1 → state)
 - heuristic – rules of thumb that seem to work in most cases
 - output – use outputs to help encode states
- No guarantee of optimality – an intractable problem

Sequential and random assignment

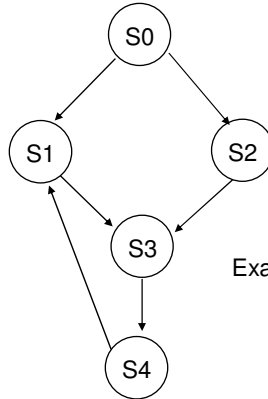
- 0..00 for the first state, 0..01 for the second, 0..10 for the third, etc.
 - For example, for two states:
 - S0=00 S1= 01 S2 = 10 S3=11
- Random assignment
 - For example, for two states:
 - S0=01 S1= 00 S2 = 11 S3=10
- Different assignment has different complexity of the next state logic and output logic

One-hot state assignment

- Simple
 - easy to encode
 - easy to debug
- Small logic functions
 - each state function requires only predecessor state bits as input
- Good for programmable devices
 - lots of flip-flops readily available
- Impractical for large machines
 - too many states require too many flip-flops
 - decompose FSMs into smaller pieces that can be one-hot encoded
- Many slight variations to one-hot
 - one-hot + all-0

Heuristics for state assignment

- Reduce the distance in Boolean n-space between related states. That is, the encodings of related states should differ by as few bits as possible



Example finite state machine

Heuristics for state assignment

- Two state assignments

State Name	Q2	Q1	Q0
S0	0	0	0
S1	1	0	1
S2	1	1	1
S3	0	1	0
S4	0	1	1

State Name	Q2	Q1	Q0
S0	0	0	0
S1	0	0	1
S2	0	1	0
S3	0	1	1
S4	1	1	1

Q2 \ Q1Q0				
	s0		s4	s3
		s1	s2	

Q2 \ Q1Q0				
	s0	s1	s3	s2
			s4	

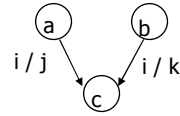
State maps: similar to K-maps, provide means of observing adjacencies in state assignments.

Up to 6 variables

Heuristics for state assignment

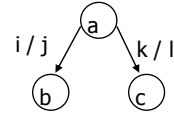
- Guidelines based on next state and input/outputs
- Highest priority: Adjacent codes to states that share a common next state
 - group 1's in next state map

I	Q	Q ⁺	O
i	a	c	j
i	b	c	k



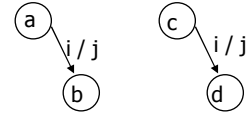
- Medium priority: Adjacent codes to states that share a common ancestor state
 - group 1's in next state map

I	Q	Q ⁺	O
i	a	b	j
k	a	c	l



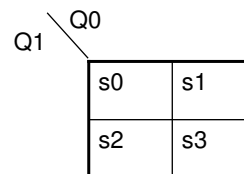
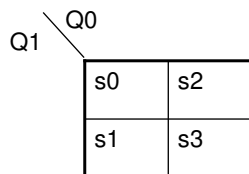
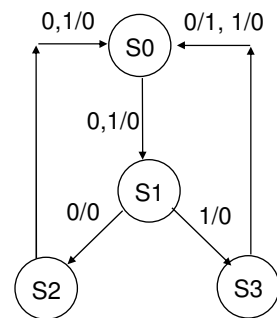
- Lowest priority: Adjacent codes to states that have a common output behavior
 - group 1's in output map

I	Q	Q ⁺	O
i	a	b	j
i	c	d	j



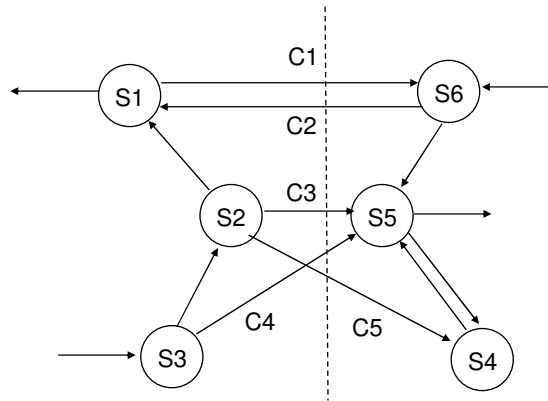
Heuristics for state assignment

- Example of 3 bit sequence detector
- Highest priority:
 - {s2, s3}
- Medium priority:
 - {s2, s3}
- Lowest priority:
 - 0/0: {s0, s1, s2}, 1/0: {s0, s1, s2, s3}



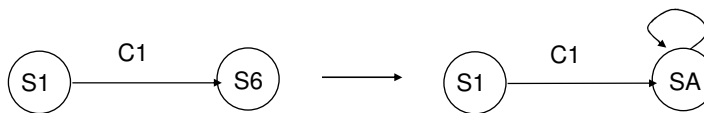
FSM partitioning

- Why FSM partitioning?
 - If an FSM is too big or complex, it can not fit in a single programmable logic component
- Partition the big FSM into two pieces that communicate with each other
- Example

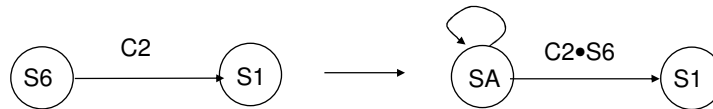


FSM partitioning

- Rules of partitioning: introducing idle state



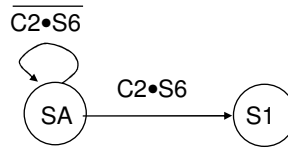
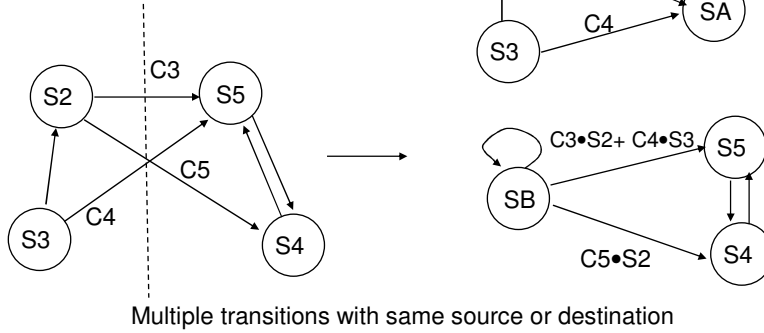
Source state transformation



Destination state transformation

FSM partitioning

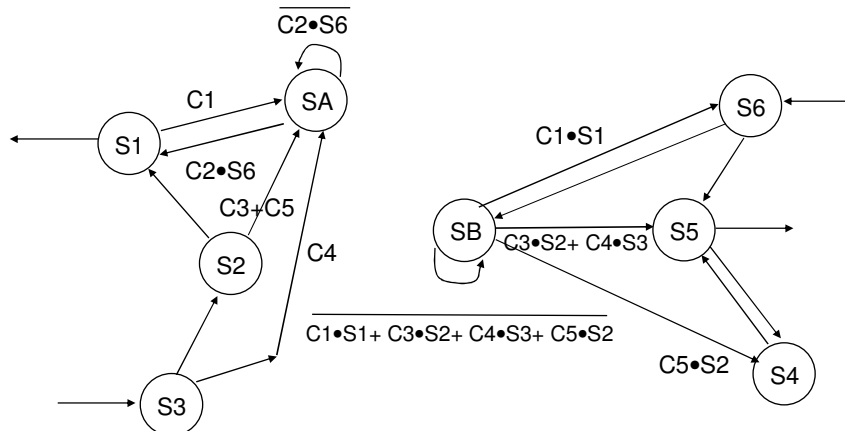
- Rules of partitioning: introducing idle state



Hold condition for idle state

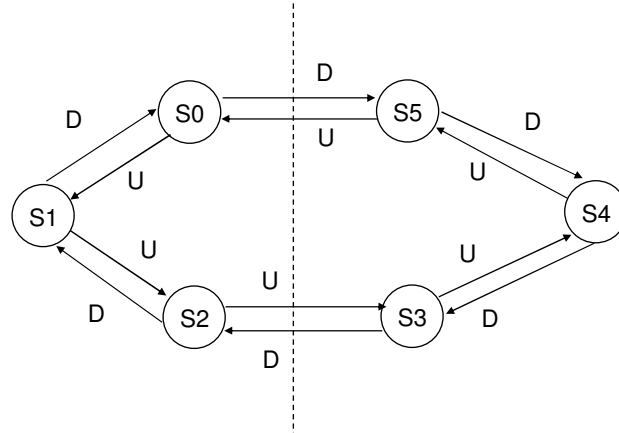
FSM partitioning

- Partitioning of the previous example



Partitioning example

- 6-state up/down counter



Summary

- State assignment
 - Sequential
 - Random
 - One-hot
 - Heuristic
- FSM partitioning
 - Why: too big to fit in some programmable logic
 - How: some rules